

Distributed ETALIS Manual

Jia Ding

23.05.2013

Contents

1	About	3
2	Setting and Build	3
2.1	Setting up a Storm cluster	3
2.1.1	Set up a Zookeeper cluster	3
2.1.2	Install dependencies on Nimbus and worker machines	3
2.1.3	Download and extract a Storm release to Nimbus and worker machines	5
2.1.4	Fill in mandatory configurations into storm.yaml file	5
2.1.5	Launch daemons under supervision using storm script and a supervisor of your choice	6
2.2	Setting up Kafka	7
2.3	Install SWI-Prolog	7
2.4	Setting up jtalis-distribute	7
2.5	Setting up distributed-Etalis	8
2.6	Run the Demo	8

1 About

Distributed-ETALIS is a distributed event processing engine. It is based on **Storm** [1] (a distributed real time computation system) and **Kafka** [2] (a distributed publish-subscribe messaging system). To set up the whole environment please follow the instruction in the next section.

2 Setting and Build

Storm cluster mode can only work under Linux, all the developments of distributed-ETALIS are under **Ubuntu** 12.04-LTS [3].

2.1 Setting up a Storm cluster

You can read the wiki page of storm (<https://github.com/nathanmarz/storm/wiki/Setting-up-a-Storm-cluster>) or read the following subsections.

2.1.1 Set up a Zookeeper cluster

Storm uses Zookeeper for coordinating the cluster. Zookeeper is not used for message passing, so the load Storm places on Zookeeper is quite low. Single node Zookeeper clusters should be sufficient for most cases, but if you want failover or are deploying large Storm clusters you may want larger Zookeeper clusters. Instructions for deploying Zookeeper are at (<http://zookeeper.apache.org/doc/r3.3.3/zookeeperAdmin.html>).

A few notes about Zookeeper deployment:

- It's critical that you run Zookeeper under supervision, since Zookeeper is fail-fast and will exit the process if it encounters any error case.
- It's critical that you set up a cron to compact Zookeeper's data and transaction logs. The Zookeeper daemon does not do this on its own, and if you don't set up a cron, Zookeeper will quickly run out of disk space.

2.1.2 Install dependencies on Nimbus and worker machines

Next you need to install Storm's dependencies on Nimbus and worker machines. These are:

- **ZeroMQ 2.1.7**

Storm has been tested with ZeroMQ 2.1.7, and is the recommended ZeroMQ release that you install. You can download a ZeroMQ release from <http://download.zeromq.org/>. Installing ZeroMQ should look something like this:

```
wget http://download.zeromq.org/zeromq-2.1.7.tar.gz
tar -xzf zeromq-2.1.7.tar.gz
cd zeromq-2.1.7
./configure
make
sudo make install
```

- **JZMQ**

JZMQ is the Java bindings for ZeroMQ. JZMQ does not have any releases, so there is risk of a regression if you always install from the master branch. To prevent a regression from happening, you should instead install from this fork which is tested to work with Storm. Installing JZMQ should look something like this:

```
git clone https://github.com/nathanmarz/jzmq.git
cd jzmq
./autogen.sh
./configure
make
sudo make install
```

To get the JZMQ build to work, you may need to do one or all of the following:

1. Set JAVA_HOME environment variable appropriately
2. Install Java dev package
3. Upgrade autoconf on your machine
4. If you run any errors when running `./configure`, this thread (<http://stackoverflow.com/questions/3522248/how-do-i-compile-jzmq-for-zeromq-on-osx>) may provide a solution.

- **Java 6**
- **Python 2.6.6**

These are the versions of the dependencies that have been tested with Storm. Storm may or may not work with different versions of Java and/or Python.

2.1.3 Download and extract a Storm release to Nimbus and worker machines

Next, download a Storm release and extract the zip file somewhere on Nimbus and each of the worker machines. The Storm releases can be downloaded from <https://github.com/nathanmarz/storm/downloads>.

2.1.4 Fill in mandatory configurations into storm.yaml file

The Storm release contains a file at `conf/storm.yaml` that configures the Storm daemons. There's a few configurations that are mandatory to get a working cluster:

- 1) **storm.zookeeper.servers**: This is a list of the hosts in the Zookeeper cluster for your Storm cluster. It should look something like:

```
storm.zookeeper.servers:
  - "111.222.333.444"
  - "555.666.777.888"
```

If the port that your Zookeeper cluster uses is different than the default, you should set **storm.zookeeper.port** as well.

- 2) **storm.local.dir**: The Nimbus and Supervisor daemons require a directory on the local disk to store small amounts of state (like jars, configs, and things like that). You should create that directory on each machine, give it proper permissions, and then fill in the directory location using this config. For example:

```
storm.local.dir: "/mnt/storm"
```

- 3) **java.library.path:** This is the load path for the native libraries that Storm uses (ZeroMQ and JZMQ). The default of `"/usr/local/lib:/opt/local/lib:/usr/lib"` should be fine for most installations, so you probably don't need to set this config.
- 4) **nimbus.host:** The worker nodes need to know which machine is the master in order to download topology jars and configs. For example:

```
nimbus.host : "111.222.333.44"
```

- 5) **supervisor.slots.ports:** For each worker machine, you configure how many workers run on that machine with this config. Each worker uses a single port for receiving messages, and this setting defines which ports are open for use. If you define five ports here, then Storm will allocate up to five workers to run on this machine. If you define three ports, Storm will only run up to three. By default, this setting is configured to run 4 workers on the ports 6700, 6701, 6702, and 6703. For example:

```
supervisor.slots.ports :  
  - 6700  
  - 6701  
  - 6702  
  - 6703
```

2.1.5 Launch daemons under supervision using storm script and a supervisor of your choice

The last step is to launch all the Storm daemons. It is critical that you run each of these daemons under supervision. Storm is a fail-fast system which means the processes will halt whenever an unexpected error is encountered. Storm is designed so that it can safely halt at any point and recover correctly when the process is restarted. This is why Storm keeps no state in-process – if Nimbus or the Supervisors restart, the running topologies are unaffected. Here's how to run the Storm daemons:

- **Nimbus:** Run the command `"bin/storm nimbus"` under supervision on the master machine.

- **Supervisor:** Run the command "bin/storm supervisor" under supervision on each worker machine. The supervisor daemon is responsible for starting and stopping worker processes on that machine.
- **UI:** Run the Storm UI (a site you can access from the browser that gives diagnostics on the cluster and topologies) by running the command "bin/storm ui" under supervision. The UI can be accessed by navigating your web browser to `http://nimbus-host:8080`.

As you can see, running the daemons is very straightforward. The daemons will log to the `logs/` directory in wherever you extracted the Storm release.

2.2 Setting up Kafka

You can download a release of Kafka from <http://kafka.apache.org/downloads.html> and unpack it on nimbus and each worker machines. The current stable version is *0.7.2-incubating*.

2.3 Install SWI-Prolog

You should follow this link (<http://www.swi-prolog.org/build/Debian.html>) to install **SWI-Prolog**. When you install it, you should remember to enable the *jpl* and *shared* flag. After you install the SWI-Prolog, you should specify the path for the **JPL** library. For example:

```
export LD_LIBRARY_PATH=/usr/local/lib/swipl-6.3.10/lib/x86_64-linux
```

2.4 Setting up jtalis-distribute

Jtalis-distribute is a distributed version of Jtalis, which is a Java wrapper for **ETALIS** [5] engine. To set up the jtalis-distribute you need to install **Subversion** and **Maven**.

You should checkout the source from <http://etalis.googlecode.com/svn/dEtalis/jtalis-distribute> and build it:

```
svn checkout http://etalis.googlecode.com/svn/dEtalis/jtalis-distribute
cd jtalis-distribute/scripts
```

```
./build.sh
```

2.5 Setting up distributed-Etalis

You should checkout the source from <http://etalis.googlecode.com/svn/dEtalis/jtalis-storm-zstream>. Then use **Maven** to build it.

```
svn checkout http://etalis.googlecode.com/svn/dEtalis/jtalis-storm-zstream
cd jtalis-storm-zstream
mvn install
```

In the target folder, you will find *jtalis-storm-zstream-0.0.1-SNAPSHOT.jar*.

2.6 Run the Demo

Follow the below instructions to run the demo:

- 1) **start zookeeper:** start a local instance of the zookeeper server in Kafka folder. For example:

```
cd kafka-0.7.2-incubating-src
sudo ./bin/zookeeper-server-start.sh config/zookeeper.properties
```

- 2) **start kafka:** start a Kafka broker

```
cd kafka-0.7.2-incubating-src
sudo ./bin/kafka-server-start.sh config/server.properties
```

- 3) **start Storm:** you should first start Nimbus and then start supervisor. At last you can start storm UI. For example:

```
cd storm-0.8.2
```



```
sudo ./bin/storm nimbus  
sudo ./bin/storm supervisor  
sudo ./bin/storm ui
```

4) **start Demo:**

```
demo.bat
```

References

- [1] <http://storm-project.net/>.
- [2] <http://kafka.apache.org/>.
- [3] <http://www.ubuntu.com/>.
- [4] <https://github.com/nathanmarz/storm/wiki/Setting-up-a-Storm-cluster>.
- [5] <http://code.google.com/p/etalis/>.